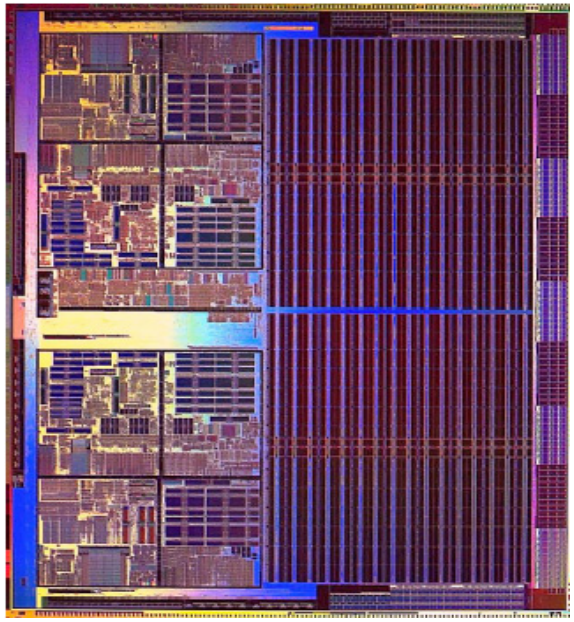
AMD™64 Virtualization

AMD India Developer's Conference
Bangalore, 10-May-2006

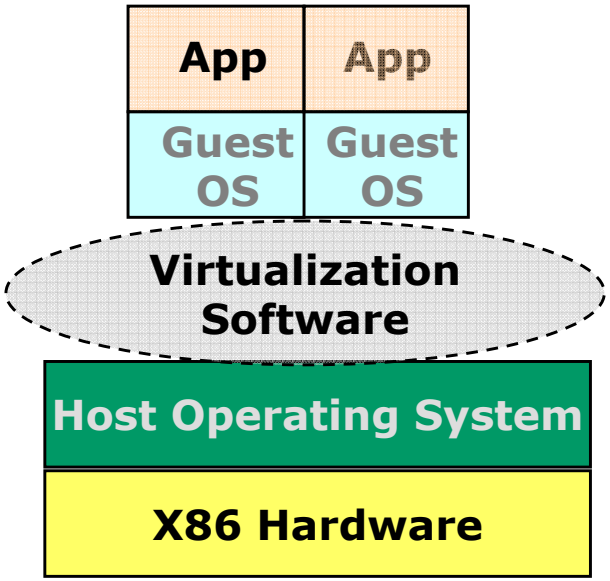
David O'Brien
Senior Systems Software Engineer
Advanced Micro Devices, Inc.



Virtual Machine Approaches

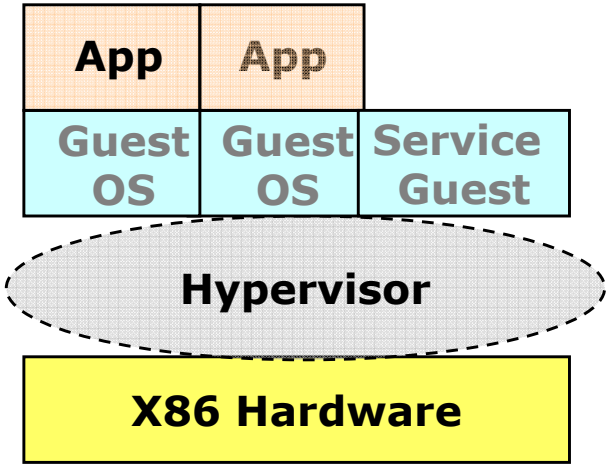
Carve a System into Many Virtual Machines

Hosted Virtualization



- Virtualization software manages resources between Host and Guest OS's
- Application can suffer decreased performance due to added overhead

Hypervisor-based Virtualization



- Virtualization Software (Hypervisor) is the host environment.
- Enables better SW performance by eliminating some of associated overhead
- If Hardware is available, the Hypervisor can be designed to take advantage of it

Driving virtualization into the processor with SVM!

- Native virtualization of x86 architecture requires "unnatural acts" to achieve – leading to increased performance overhead & complexity
- Moving functionality traditionally served by software-based hypervisor into the processor helps to solve these problems.
- ***SVM – Secure Virtual Machine (PACIFICA) is next logical evolution to the AMD's Direct Connect Architecture* to provide technology for silicon enhanced virtualization**
- SVM allows the software vendors to focus on the value-add, leaving the worry of proper emulation to the processor.

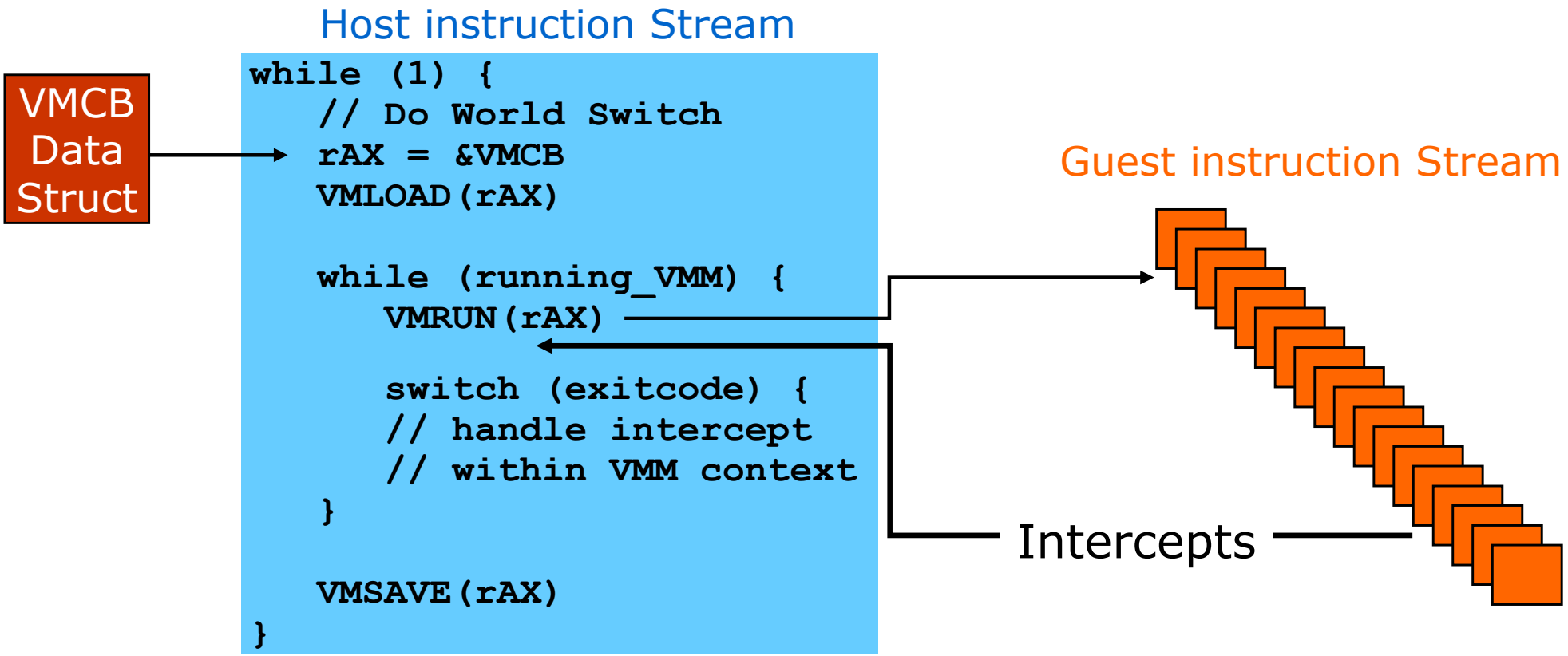
SVM virtualization technology allows AMD to continue to offer a competitive performance roadmap while meeting the system architecture demands of our customers

SVM Overview & Highlights

- SVM drastically reduces the complexity and performance impact of existing x86/64 virtualization
- SVM enabled parts will launch in AMD processors beginning in 2006 across segments; mobile, server/workstation, and desktop markets
- Compatible with x86 and AMD64 applications – no change in legacy software is required.
  - Allows hypervisor to simultaneously support multiple Guests in any mode – 16-bit, 32-bit, 64-bit
  - Works with multi-processor/multi-core host platforms
  - Allows hypervisor to support SMP-aware guests.

Core "SVM" Architecture: VMRUN

- Virtualization based on Virtual Machine Run (VMRUN) instr.
- VMRUN executed by HV causes guest to run in "Guest Mode"
- Guest runs until it exits back to the hypervisor
- Hypervisor resumes at the instruction following VMRUN
- World-switch: hypervisor → guest → hypervisor



Core "SVM" Architecture: Intercepts

- Guest runs until:
  - It performs an action that causes an exit to the host
  - It explicitly executes the `VMMCALL` instruction
- The VMCB for a guest has settings that determine what actions cause the guest to exit to host
  - These intercepts can vary from guest to guest
  - Two kinds of intercepts
    - *Exception & Interrupt Intercepts*
    - *Instruction Intercepts*
  - Rich set of intercepts allow the host to set customize each guest's privileges
- Information about the intercepted event is put into the VMCB on exit

SVM Architecture Features

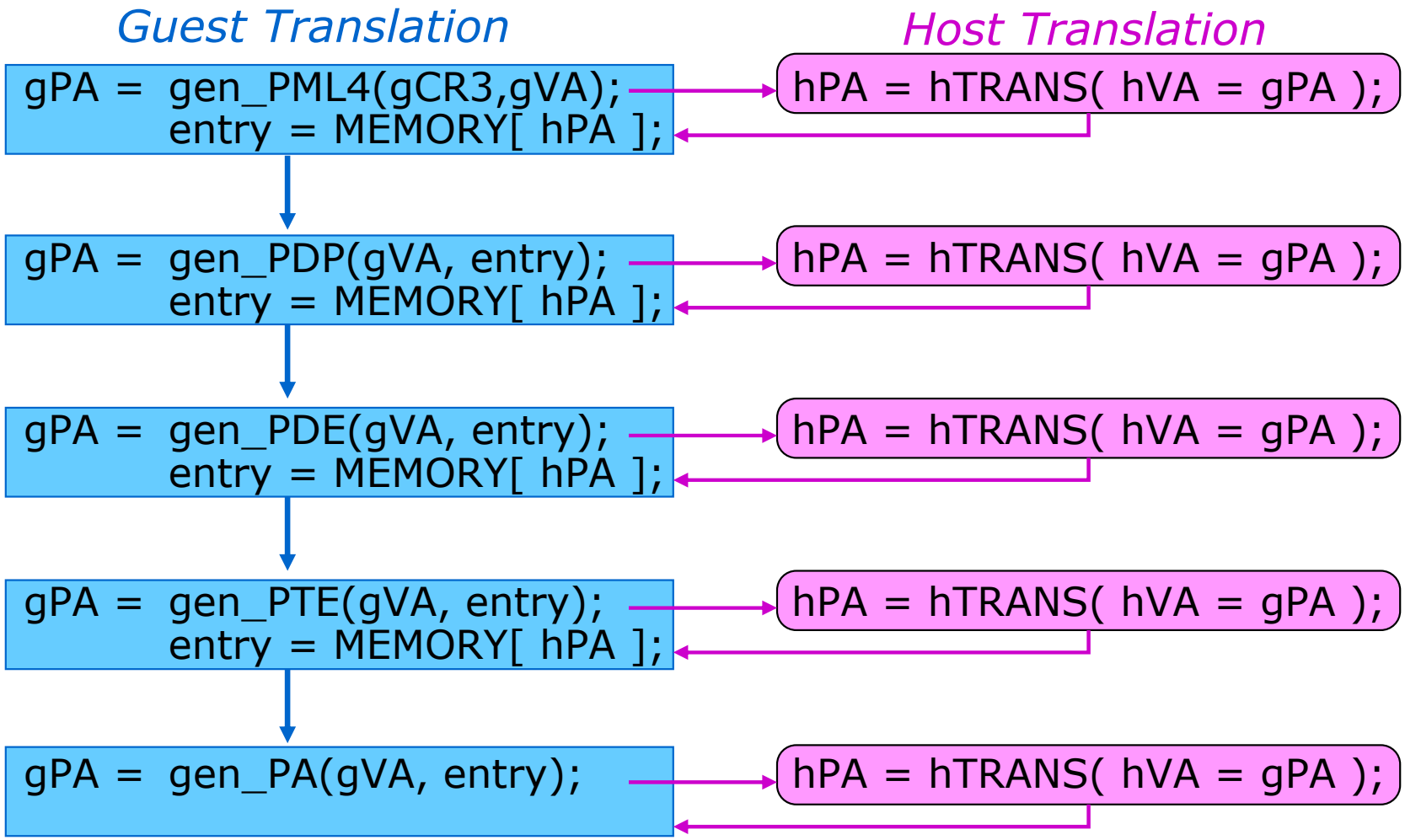
- New Instruction Set Features
  - Processor Mode: Guest Mode
  - Data Structure: Virtual Machine Control Block (VMCB), etc
  - Instructions: VMRUN, VMLOAD, VMSAVE, VMMCALL, etc
  - All instructions now Restartable
- Interrupt architecture changes
  - Selective Interception:
    - *increasing performance and enabling para-virtualization*
  - Event Injection
    - *Eliminates need for VMM code to emulate x86 exception delivery*
    - *Designed to reduce VMM development time significantly*
- Paging Features: (See next slide)
- Security Features: DMA Exclusion Vectors, SKINIT, etc

SVM Paging Support

- New Paging Features & Support:
  - Shadow Page Tables (SPT) & Nested Page Tables
  - New memory mode: Real Mode w/ Paging
- "Tagged TLB": Address Space ID's (ASID) added to improve Translation Look-aside Buffer (TLB) performance
  - VMRUN sets guest ASID
- Shadow Paging Support
  - Host intercepts guest CR3 Reads/Writes
  - Host monitors guest edits to guest page tables which are marked "read only" by host
  - Host constructs and manages SPT in software
  - Guest never sees "real" page tables or real content of CR3

Nested Paging: CPU walks both Guest & Host Page Tables

- CPU maps each Guest_PA to Host_VA and then translates to Host_PA
- CPU builds compound gVA_to_hPA TLB entries (guarded by ASID)
- Far more efficient than "Shadow Page Tables", all handled by CPU



Trademark attribution and cautionary statement

CAUTIONARY STATEMENT

This presentation contains forward-looking statements, which are made pursuant to the safe harbor provisions of the Private Securities Litigation Reform Act of 1995. Forward-looking statements in this presentation relates to, among other things, product and technology introduction schedules; and planned product architecture features. Forward-looking statements involve risks and uncertainties that could cause actual results to differ materially from the company's current expectations. Risks include the possibility that the company may not achieve its current product and technology introduction schedules; that global business and economic conditions will worsen resulting in lower than currently expected sales; that Intel Corporation's pricing, marketing programs, product bundling, new product introductions or other activities targeting the company's processor business will prevent attainment of the company's current plans; that demand for personal computers and, in turn, demand for the company's processors, will be lower than currently expected; that adoption of AMD64 products by OEMs will not continue as expected; that the company will not be able to meet demand for its products; that the company will not be able to raise sufficient capital to enable it to establish leading-edge capacity to maintain its market leadership positions; and that solutions providers will not timely provide the infrastructure to support the company's AMD64 technology.

Because the company's actual results may differ materially from our plans and expectations today, we encourage you to review the company's filings with the Securities and Exchange Commission, including but not limited to the risks and uncertainties contained in our Annual Report on Form 10-K for the year ended December 26, 2004, and our Quarterly Report on Form 10-Q for the quarter ended March 27, 2005.